

## LA-UR-14-27640

Approved for public release; distribution is unlimited.

Title: Learning Tree-structured Approximations for Conditional Random Fields

Author(s): Skurikhin, Alexei N.

Intended for: the Applied Imagery Pattern Recognition (AIPR) annual workshop,  
2014-10-14/2014-10-16 (Washington DC, District Of Columbia, United  
States)

Issued: 2014-11-18 (rev.3)

---

**Disclaimer:**

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

# Learning Tree-structured Approximations for Conditional Random Fields

Alexei N. Skurikhin

Intelligence Space and Research Division  
Los Alamos National Laboratory  
Los Alamos, NM 87545, USA  
alexei@lanl.gov

**Abstract**—Exact probabilistic inference is computationally intractable in general probabilistic graph-based models, such as Markov Random Fields and Conditional Random Fields (CRFs). We investigate spanning tree approximations for the discriminative CRF model. We decompose the original computationally intractable grid-structured CRF model containing many cycles into a set of tractable sub-models using a set of spanning trees. The structure of spanning trees is generated uniformly at random among all spanning trees of the original graph. These trees are learned independently to address the classification problem and Maximum Posterior Marginal estimation is performed on each individual tree. Classification labels are produced via voting strategy over the marginals obtained on the sampled spanning trees. The learning is computationally efficient because the inference on trees is exact and efficient. Our objective is to investigate the capability of approximation of the original loopy graph model with loopy belief propagation inference via learning a pool of randomly sampled acyclic graphs. We focus on the impact of memorizing the structure of sampled trees. We compare two approaches to create an ensemble of spanning trees, whose parameters are optimized during learning: (1) memorizing the structure of the sampled spanning trees used during learning and, (2) not storing the structure of the sampled spanning trees after learning and regenerating trees anew. Experiments are done on two image datasets consisting of synthetic and real-world images. These datasets were designed for the tasks of binary image denoising and man-made structure recognition.

**Keywords**—conditional random field; belief propagation; inference; spanning tree

## I. INTRODUCTION

Complex event and object recognition requires analysis of massive quantities of data to detect and identify static or dynamic patterns. One challenge for automated object recognition, and machine learning in general, arises from the fact that in spite of growing quantities of data, object recognition often remains ill-posed, particularly for low signal-to-noise ratios. The object recognition ambiguity can be addressed by exploiting the fact that events and objects rarely occur in isolation; they tend to co-occur and co-vary, and this correlation structure, known as context, provides important information for the disambiguation of the event (Fig.1). This problem of object disambiguation is extremely common in a broad spectrum of applications, such as computer vision, social

network analysis, bioinformatics, text analytics, and the semantic web. For example, classification of Web documents can be improved by augmenting analysis of the Web page text with the page hyperlinks that define relations to other Web pages. From a broader perspective, recognition and disambiguation of interrelated objects can be posed as structured machine learning. The goal is learning structured hypothesis from data with internal structure in the form of one or more relations and global constraints in the problem input and output domains, a set of observables  $X$  and a set of output random variables  $y_i \in Y$ , respectively. The variable  $y_i$  is assumed to take a value from a finite label set  $S_i = \{s_1, s_2, \dots, s_{m_i}\}$  with cardinality  $m$ . For example, in a case of binary classification the variables have two states,  $y_i = \{0,1\}$  and may correspond to object classes such as images of man-made objects and images of nature scenes. With structure present in the problem output domain, which consists of multiple interdependent labels  $y_i \in Y$  that have to be assigned to objects, recognition of objects is done by jointly labeling all the output variables simultaneously. In such an approach each object or object component acts as context for others by mutually facilitating and constraining interpretation of each other. This collective classification exploits interdependence between the output variables and is in contrast to “standard” methods of machine learning, where input observed instances are mapped to labels independently of each other.

Undirected probabilistic graphical models such as Markov Random Fields (MRFs) and Conditional Random Fields (CRFs) are being increasingly used to model problems having a structured domain and to enable probabilistic inferences such as answering queries about the variables of interest (e.g., labeling of output variables). A key task of probabilistic inference is to compute the probability distribution over the unobserved variables  $y_i \in Y$  given the observed values of other random variables (the evidence) and accounting for prior beliefs.

There is one overarching problem for structured machine learning in undirected graphical models: developing computationally tractable learning and inference methods. Challenges are due to the computational intractability of exact inference over general graphs with many variables which is often the case in most real-world scenarios. Hence, there is a need to find ways to reduce the cost of inference both at



Fig. 1. An illustration of context priming for recognition: (a) with only information from a small region, the object is not recognizable (left), by adding the object's context, recognizing the car becomes possible (right); (b) what may appear as spiders (left), when viewed alone out of the surrounding context, becomes the top burner grates for gas range, when viewed in the context of the whole scene (right).

learning time and at run time. In this work, we focus on the problem of approximation of learning and inference in grid-structured CRF-based probabilistic model via ensemble of randomly generated spanning trees.

The CRFs are the discriminative models [1]. While MRF models the joint distribution  $P(\mathbf{X}, \mathbf{Y})$ , CRF directly models a conditional distribution  $P(\mathbf{Y}|\mathbf{X})$  and avoids modeling the distribution over the variables in  $\mathbf{X}$ ,  $P(\mathbf{X})$ . The CRF-based approach allows one to relax the assumption of conditional independence of the observed data often used in generative approaches, such as MRF, and makes it possible to incorporate almost arbitrary feature vector representations of the observed data points thus facilitating complex context modeling. The CRF model with strictly positive probability distribution can be parameterized in the form of an exponential family. This model is factorized according to a graph  $\mathbf{G}$  consisting of a set of nodes,  $\mathbf{V}$ , corresponding to random variables  $y_i$ , and a set of undirected edges,  $\mathbf{E}$ . By the Hammersley-Clifford theorem [2][3] the conditional distribution can be given in a factored form as:

$$P(\mathbf{Y}|\mathbf{X}; \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{X}; \boldsymbol{\theta})} \exp(-U(\mathbf{Y}, \mathbf{X}, \boldsymbol{\theta})) = \frac{1}{Z(\mathbf{X}; \boldsymbol{\theta})} \prod_{c=1}^C \Phi_c(\mathbf{Y}_c, \mathbf{X}_c; \boldsymbol{\theta}), \quad (1)$$

where  $\boldsymbol{\theta}$  are model parameters;  $Z(\mathbf{X}; \boldsymbol{\theta})$  is a normalization factor also known as the partition function and is, in general, intractable to compute;  $U(\mathbf{Y}, \mathbf{X}, \boldsymbol{\theta})$  is the energy function, which is the sum of clique potentials over all possible maximal cliques,  $C$ , in the graph;  $\Phi_c(\mathbf{Y}_c, \mathbf{X}_c; \boldsymbol{\theta})$  are positive factors corresponding to maximal cliques in the graph, where a maximal clique of a graph is a fully connected subset of nodes that cannot be further extended. Factors are also referred as potential functions or potentials. The normalization factor is computed by summing over all possible assignments of  $\mathbf{Y}$  in the joint label space of all nodes  $\Omega = S_1 \times S_2 \times \dots \times S_{|\mathbf{V}|}$ :

$$Z(\mathbf{X}; \boldsymbol{\theta}) = \sum_{\mathbf{Y}' \in \Omega} \prod_{c=1}^C \Phi_c(\mathbf{Y}'_c, \mathbf{X}_c; \boldsymbol{\theta}), \quad (2)$$

Thus, CRF may be viewed as an MRF globally conditioned on the observed data. In other words, CRF makes independence assumptions among  $\mathbf{Y}$ , but not among  $\mathbf{X}$ , that is when  $\mathbf{Y}$  is conditioned on  $\mathbf{X}$ , conditional distribution of  $y_i$  given its neighbors, defined by the graph  $\mathbf{G}$ , does not depend on other variables  $y_j$  outside the neighborhood of  $y_i$ .

It is generally assumed that factors are represented using a log-linear model and parameterized by means of an inner product between the parameter vector  $\boldsymbol{\theta}$  and a feature function that takes into account information relevant to the application task. In this case the factor parameterization takes the form:

$$\Phi_c(\mathbf{Y}_c, \mathbf{X}_c; \boldsymbol{\theta}) = \exp\{\boldsymbol{\theta}_c^T \cdot f_c(\mathbf{Y}_c, \mathbf{X}_c)\}, \quad (3)$$

where  $f_c$  is known as feature functions of factor  $c$ . In applications, e.g. computer vision, a pairwise CRF model is often used. This model takes into consideration only unary and pairwise cliques. The distribution is then defined as:

$$P(\mathbf{Y}|\mathbf{X}; \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{X}; \boldsymbol{\theta})} \prod_{i \in \mathbf{V}} \psi_i(\mathbf{Y}_i, \mathbf{X}_i; \boldsymbol{\theta}_i) \prod_{(i,j) \in \mathbf{E}} \psi_{ij}(\mathbf{Y}_i, \mathbf{Y}_j, \mathbf{X}_i, \mathbf{X}_j; \boldsymbol{\theta}_{ij}), \quad (4)$$

where  $\psi_i$  and  $\psi_{ij}$  are clique factors. In the context of the classification task,  $\psi_i$  is the association potentials defined over the node that measures the support for class label  $y_i$  of node  $i$ , and the pairwise potentials  $\psi_{ij}$  defined over graph edges are the interaction potentials that represent “compatibility” between the different label assignments to the nodes  $i$  and  $j$ . In computer vision tasks, for instance, the node  $y_i$  can represent a random variable that is a label corresponding to an individual pixel patch and  $x_i$  is a feature vector containing color characteristics of this pixel patch. Due to lack of restrictions on  $x_i$ , it can include characteristics of neighboring patches as well. In the example above, the interaction potential might estimate similarity of neighboring pixel patches in terms of their color features as well as compatibility of their labels.

Learning the CRF model corresponds to finding the model parameters,  $\boldsymbol{\theta}^*$ , that maximize the conditional log-likelihood objective,  $L$ , on the training data  $D$ :

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} L(D, \boldsymbol{\theta}) = \arg \max_{\boldsymbol{\theta}} \left( \frac{1}{|D|} \sum_{(\mathbf{X}, \mathbf{Y}) \in D} \log P(\mathbf{Y}|\mathbf{X}; \boldsymbol{\theta}) - \lambda \cdot \|\boldsymbol{\theta}\|^2 \right), \quad (5)$$

where the regularization penalty term  $\lambda \cdot \|\boldsymbol{\theta}\|^2$  ( $\lambda = 1/2\sigma^2$ ) is a Gaussian prior imposed on the parameters to control for overfitting. Estimation of the gradient of the log-likelihood objective requires computing marginals and is usually not tractable due to the presence of the partition function. Once the model is learned, inferring classification labels is done using either maximum a posteriori (MAP) or the maximum posterior marginals (MPM) criteria that, similar to

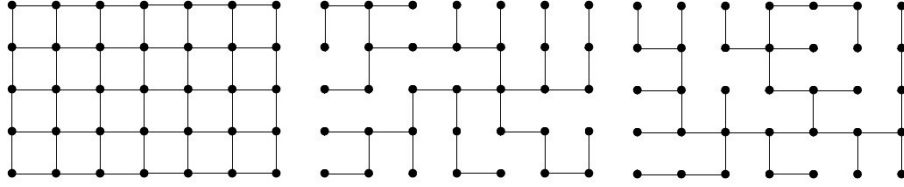


Fig. 2. Illustration of two spanning trees corresponding to a  $5 \times 7$  grid graph. (Left) Original graph, (Center and Right) two randomly generated spanning trees.

learning, require computing marginals and the partition function. Computationally efficient inference is crucial for learning as the inference has to be done many times in the course of training.

There are several cases when inference is tractable. They include tree structured graphs [4], graphs with low treewidth [5,6,7,8], and binary pairwise Markov Random Fields (MRFs) with submodular energy constraints [9]. The complexity arises due to the intractability of inference in general graphs for which it is known to be *NP*-hard [10, 11, 12]. Different approaches to approximate inference have been proposed. The pseudo-likelihood approach approximates the true likelihood objective with a factorization of the local conditional likelihoods [13, 14]. Piecewise pseudo-likelihood takes the idea of pseudo-likelihood further by decomposing a graphical model into pieces, each of which is trained separately and then combined into a model [15, 16]. The blocked contrastive divergence approach uses sampling of tree-structured blocks to approximate the gradient of the likelihood with the gradient of the composite likelihood [17].

A frequently used approach for approximate inference is loopy belief propagation (LBP), in which belief propagation is applied to the graph, even if it is not a tree [4]. However, LBP is not guaranteed to converge and when it converges the result is not guaranteed to be a unique fixed point. One of the well-known approaches to approximate the inference is tree-reweighted message passing (TRW) that decomposes the MAP estimation problem over a loopy graph into tractable subproblems over acyclic subgraphs and then employs message passing over trees to combine solutions [18]. The original TRW approach was modified into sequential tree-reweighted message passing scheme (TRW-S) that addressed the lack of convergence in the TRW and tightened the lower bound on the energy function [19].

Another approach to approximate the intractable loopy graph model is via a mixture of spanning trees [20]. A set of randomly sampled spanning trees is generated (Fig. 2). These trees are learned separately to address the overall classification problem and inference is then performed on each individual tree. The learning is computationally efficient because the inference on trees is exact and efficient. Classification labels can be produced via voting strategy over MAP labels obtained on the sampled spanning trees. In this work, we empirically investigate the impact of memorizing the structure of the sampled spanning trees used during learning. We compare two approaches to create an ensemble of spanning trees, whose parameters are optimized during learning: (1)

memorizing the structure of the sampled spanning trees used during learning and, (2) not storing the structure of the sampled spanning trees after learning. In the first approach, during test phase we create an ensemble by reusing spanning trees with the memorized structures and corresponding optimized sets of tree parameters. In the second approach, we randomly regenerate spanning trees anew and use the same set of optimized parameters for all trees. Experiments were done on two image datasets consisting of synthetic and real-world images.

## II. CRF TREE-STRUCTURED APPROXIMATION

### A. Inference in tree-structured graphs

Inference is a fundamental problem that arises during both the test and training phases. During training, computing the gradient requires inference for the computation of node and edge marginals, which can be done efficiently for tree structured graphs or graphs with small treewidth. Inference in tree-structured graphs can be implemented efficiently using belief propagation that is an iterative message passing algorithm for computing approximate marginal distributions of each variable in a graphical model [4]. The algorithm is based on the idea of message passing along the edges of the graph, where the message indicates how much each node should update its belief based on the neighboring node's current information. In tree-structured graphs the message passing schedule can be considered as a two-pass schedule, with forward message propagation followed by backward propagation. The algorithm begins by defining one of the variables (nodes) as the root. In the forward sweep the algorithm sends messages from leaves towards the root. In the backward sweep the messages are propagated in the opposite direction, from the root through tree edges to all leaves. After the completion of the backward sweep, the full set of messages over every edge in both directions is estimated. This allows one to compute all the node and edge (joint) marginals. The computational complexity of the algorithm is  $O(Nm^2)$ , where  $N=|V|$  is the number of node variables, and  $m = \max_i m_i$ . In loopy graphs, neither convergence nor correctness of this procedure is guaranteed as the message of one node can pass back to itself through the loops; thus approximate methods, such as LBP, can be used.

The important question is how to choose trees to approximate the original graphical model. The number of spanning trees of the graph is exponential in the size of the graph. The exact number of spanning trees can be estimated using the matrix-tree theorem by evaluating the

determinant of the combinatorial Laplacian of the graph. We take an approach that assumes a lack of prior information on more optimal graph partitioning, or that complex objects which have to be recognized do not show any specific patterns in spatial arrangements of their components, except possibly being closer to each other. It is similar to a very general scenario of context-driven recognition, when co-occurrence of individual components facilitates recognition of each component by the others. In the end, collective recognition of the components facilitates the recognition of the whole object they constitute. Therefore, we choose to randomly generate spanning trees. Each of trees captures some aspect of the original model. The use of randomly sampled trees allows us to compensate to some extent for weakness of each of the individual spanning trees. Spanning trees are generated uniformly at random from among all spanning trees of the original graph using the algorithm, introduced in [21], that simulates a loop-erased random walk. The complexity of this generation is  $O(\bar{\tau})$ , where  $\bar{\tau}$  is the mean hitting time of a graph, which can be much smaller than the cover time of a graph. In the case of memorizing the structure of spanning trees, we generate a predefined number of trees and then train every generated tree on the whole training dataset. Classification is done by majority voting over the pool of these stored trained trees. In the case of not storing the structure of spanning trees, we randomly generate a pool of spanning trees either anew for each data sample of the training and test datasets, or we generate the pool once for training and then randomly generate new pool before testing begins.

### B. Learning

For the tree-structured model efficient exact parameter learning is possible. Parameters of every spanning tree are optimized by maximizing the log-likelihood  $L$  on the training data  $D$ . This optimization can be performed by gradient-based methods. Using the log-linear model of factors in equation (4) the conditional log-likelihood in equation (5) can be rewritten as

$$L = \frac{1}{|D|} \sum_{(X,Y) \in D} \left( \log \frac{1}{Z(X; \theta)} \prod_{c=1}^C \exp\{\theta_c^T f_c(Y_c, X_c)\} \right) - \lambda \cdot \|\theta\|^2 \quad (6)$$

$$L = \frac{1}{|D|} \sum_{(X,Y) \in D} \sum_{c=1}^C \theta_c^T f_c(Y_c, X_c) - \frac{1}{|D|} \sum_{(X,Y) \in D} \log Z(X; \theta) - \lambda \cdot \|\theta\|^2, \quad (7)$$

Differentiating the log-likelihood with respect to  $\theta$  and using equation (2), the partial derivative of the log-likelihood with respect to  $\theta$  is expressed as

$$\frac{\partial L}{\partial \theta} = \frac{1}{|D|} \sum_{(X,Y) \in D} f_c(Y_c, X_c) - \frac{1}{|D|} \sum_{(X,Y) \in D} \frac{1}{Z(X; \theta)} \frac{\partial Z(X; \theta)}{\partial \theta} - \lambda \cdot \|\theta\|, \quad (8)$$

The partial derivative of the second term with respect to the parameter  $\theta$  is computed as

$$\frac{1}{|D|} \sum_{(X,Y) \in D} \frac{1}{Z(X; \theta)} \sum_{Y' \in \Omega} \left( \prod_{c=1}^C \exp\{\theta_c^T f_c(Y'_c, X_c)\} \right) \cdot \sum_{c=1}^C f_c(Y'_c, X_c) \quad (9)$$

$$= \frac{1}{|D|} \sum_{(X,Y) \in D} \frac{1}{Z(X; \theta)} \sum_{Y' \in \Omega} P(Y'|X) \cdot \sum_{c=1}^C f_c(Y'_c, X_c), \quad (10)$$

The first term in (8) is the expected value of  $f$  under the empirical distribution and the partial derivative of the second term is the expectation under the model distribution. Thus, the derivative of the conditional log-likelihood with respect to a parameter  $\theta$  can be expressed as a difference between the expected and observed feature responses. The sums of the feature values  $f$  in (8) and (10) are also known as sufficient statistics.

Computation of the gradient requires the marginal probabilities,  $P(Y'|X)$ , which can be efficiently computed in trees using the forward-backward algorithm discussed in the previous section. The model parameters are initialized using maximum likelihood estimation for the logistic regression model. Then, we use a limited memory quasi-Newton method, such as L-BFGS [22], to optimize the parameters of the tree-structured CRF model.

When we memorize the structure of the spanning trees, for every kept tree we optimize a corresponding set of parameters. When we do not store the structure of spanning trees, we optimize one set of parameters that is used by all the spanning trees in the pool.

## III. PERFORMANCE EVALUATION

We evaluate spanning tree structured approximations for CRF on a number of synthetic and real-world image datasets and compare it with the results presented in [20,23,24,25], as well as with the results produced using a logistic regression classifier and the grid-structured CRF model with LBP inference. We use the same two datasets that were used in [23,24]. Regularization of CRF node and edge parameters is done with the corresponding values of the regularization parameters  $\lambda_n, \lambda_e$ . We search for  $\lambda_n, \lambda_e$  in a grid like fashion on pairwise combinations  $\lambda_n, \lambda_e$  which are formed using a set of allowable  $\lambda = \{0.001, 0.01, 0.1, 1, 10, 100\}$ . We used one of the folds in the training dataset corresponding to the problem of binary image denoising that is presented in the next section. Then, the same values of  $\lambda_n, \lambda_e$  were used in the second experiment on real-world images. We used  $\lambda_n=0.001$  and  $\lambda_e=100$ , though other combinations were comparable to the chosen one in terms of pixelwise classification error. In all our experiments we initialize the model parameters with zeroes and find starting estimates of the node parameters using a logistic regression model, which are then used in the follow-up training of the full model, e.g. in the grid-structured CRF model with LBP inference.

### A. Synthetic images: binary image denoising

We used the binary image denoising dataset that is available in [23,24]. The dataset includes two subsets containing images corrupted with either unimodal or bimodal Gaussian noise. There are fifty  $64 \times 64$  pixels noisy image samples corresponding to one of the four reference

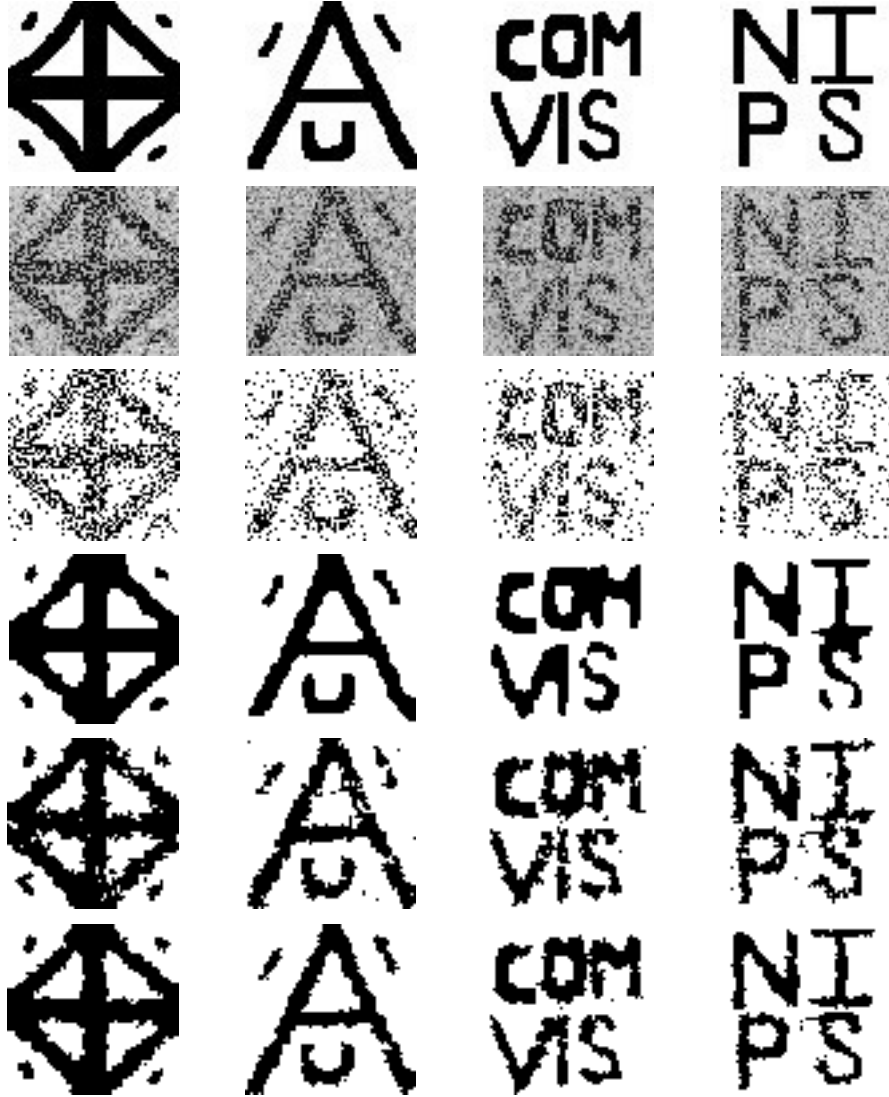


Fig. 3. Results of binary denoising task. From top: row 1: reference images, row 2: images corrupted with bimodal noise, row 3: logistic classifier results, row 4: results of the grid structured CRF model with loopy belief propagation inference, row 5: result using one spanning tree; row 6: result using a majority voting on a set of 15 CRF spanning trees with randomly generated structures and the same set of optimized (learned) parameters.

TABLE I. PIXELWISE CLASSIFICATION ERROR (%) FOR BINARY DENOISING TASK. KH'06 STANDS FOR THE RESULTS PUBLISHED IN [23, 24]. COMPARISONS TO LOGISTIC CLASSIFIER AND GRID-STRUCTURED CRF MODEL WITH LOOPY BELIEF PROPAGATION INFERENCE (LBP) ARE ALSO SHOWN. FOR LBP INFERENCE CASE WE AVERAGE OVER FIVE DIFFERENT RUNS EACH ON 160 TEST SAMPLES. MEAN  $\pm$  STANDARD DEVIATION IS SHOWN FOR OUR RESULTS AND POB'09 RESULTS THAT ARE AVERAGED OVER FIVE RUNS TOO

	Unimodal	Bimodal
Logistic regression based classifier	14.72 $\pm$ 0.02	23.10 $\pm$ 0.04
Markov Random Field (KH'06)	2.35	7.00
Discriminative Random Fields (KH'06)	2.30	6.21
LBP (MPM estimates)	2.65 $\pm$ 0.11	6.04 $\pm$ 0.09
Majority voting on a set of spanning trees (MPM estimates, a set of 15 spanning trees), tree structure is memorized	3.37 $\pm$ 0.01	5.81 $\pm$ 0.06
Majority voting on a set of spanning trees (MPM estimates, a set of 15 spanning trees), tree structure is not memorized	3.38 $\pm$ 0.04	5.80 $\pm$ 0.02

images. Therefore, each data subset contains 200 images. The unary and pairwise features are defined similar to [23, 24]. We used  $f_i^{node} = [1, I_i]$  and  $f_{ij}^{edge} = [1, |I_i - I_j|]$ , where  $I_i$  is the pixel “ $i$ ” intensity. We used slightly modified 5-fold cross-validation, in which we train the model on one fold and test the model on the remaining  $k-1$  folds. Each fold has 40 images consisting of 10 images corresponding to each reference image. Therefore, we report average performance over five runs each on 160 test images. The results for the task of denoising images corrupted with bimodal noise are shown in Figure 3.

Similar statistics was obtained for the task of denoising images corrupted with unimodal Gaussian noise. Comparison of rows 4 and 6 in the Figure 3, as well as the summary in Table 1, show that the tree-structured approximation for CRF produces comparable level of performance with the grid-structured CRF model that uses LBP inference. Results produced when the structure of trees is memorized and when it is not, are of comparable quality.

#### B. Real-world images: structure detection

We used the dataset that was presented in [23, 24]. The training and testing datasets contain 108 and 129 images respectively, each of size  $256 \times 384$  pixels, from the Corel image database. Each image was divided into non-overlapping patches of  $16 \times 16$  pixels. The goal was to label each patch as structured (man-made) or non-structured (natural background). There are 30710 “non-structured” pixel patches and 10762 “structured” pixel patches in the training dataset. The testing dataset contains 43164 “non-structured” patches and 6372 “structured” patches.

Each pixel patch was originally characterized with a 14-dimensional feature vector computed using orientation and gradient magnitude based features similar to as described in [24]. The feature vector contains a combination of single-scale and multi-scale features. The number of scales was chosen to be 3. Multi-scale features of the pixel patch were extracted from orientation histograms of blocks sizes  $16 \times 16$ ,  $32 \times 32$ , and  $64 \times 64$  centered on the lowest scale block that covers the  $16 \times 16$  pixel patch. For each image patch, the gradients at pixels contained in the blocks are used to compute histograms of gradient orientations. Each count in the histograms is

weighted by the gradient magnitude at that pixel. The histograms were smoothed using kernel smoothing. From the smoothed histogram the first and third heaved central shift moments were extracted. Two additional features related to each scale were computed as well. The first is  $|\cos(p_1)|$  and the second is given by  $|\sin(p_1 - p_2)|$ , where  $p_1$  is the location of the highest peak of the orientation histogram, and  $p_2$  is the location of the second highest peak of the histogram. This feature favors the presence of near right-angle junctions, which are more likely to be associated with man-made structures. 60 bins were used to encompass the angular range of  $[-\pi, \pi]$ . Two interscale features were computed to facilitate detection of continuing line or near right angle present at multiple scales. These features were computed between different scales as  $|\cos 2(p_1^i - p_1^j)|$ , where  $p_1^i$  is the location of the highest peak at scale  $i$ . The differences between values corresponding to the first and second and the first and third scales were used. Then, this 14-dimensional feature vector is expanded with quadratic mapping into a 119-dimensional feature vector corresponding to each pixel patch. Pairwise feature vectors were obtained by concatenating the expanded patch feature vectors.

We compared the performance of the two approximation approaches based on an ensemble of spanning trees with the results presented in [23, 24, 25] and the grid-structured CRF model with LBP inference. We note that the false positive rate reported in [23, 24] does not count as false positive (FP) a misclassification that is adjacent to a block with ground truth labeled “structured.” In the results shown in Table 2 and in Figure 4, we follow the convention in [23, 24]. It is worth mentioning that the majority of false negatives correspond to small scale objects or flat surfaces (e.g., smooth roofs and walls), which is a matter of designing better features, better labeling and partitioning of the training images. The precision obtained with the grid-structured CRF model and LBP inference is 74.64% over the test dataset. The approximation based on majority voting on a set of 15 spanning trees without memorizing tree structure has a precision of 83.24%, while the approximation that memorizes tree structure has a precision of 79.95%. The approximation based on a cascade of spanning trees gives a precision of 79.27% [25]. Thus the last three

TABLE II. DETECTION RATES (DR) AND FALSE POSITIVE RATES (FP) FOR STRUCTURE DETECTION PROBLEM OVER THE TEST SET CONTAINING 129 IMAGES. KH’06 STANDS FOR THE RESULTS PUBLISHED IN [23,24,25]. COMPARISONS OF MAJORITY VOTING ON A SET OF 15 SPANNING TREES AND GRID-STRUCTURED CRF MODEL WITH LOOPY BELIEF PROPAGATION INFERENCE (LBP) ARE ALSO SHOWN

	DR (%)	FP (per image)
Markov Random Field (KH’06)	58.35	2.44
Discriminative Random Fields (KH’06)	72.54	1.76
LBP (MPM estimates)	85.30	14.32
Majority voting on a set of 15 spanning trees (MPM estimates), tree structure is memorized	88.01	10.90
Majority voting on a set of 15 spanning trees (MPM estimates), tree structure is not memorized	90.52	9
MPM estimates using a hierarchical cascade of spanning trees (SK’14), tree structure is not memorized	91.75	11.85

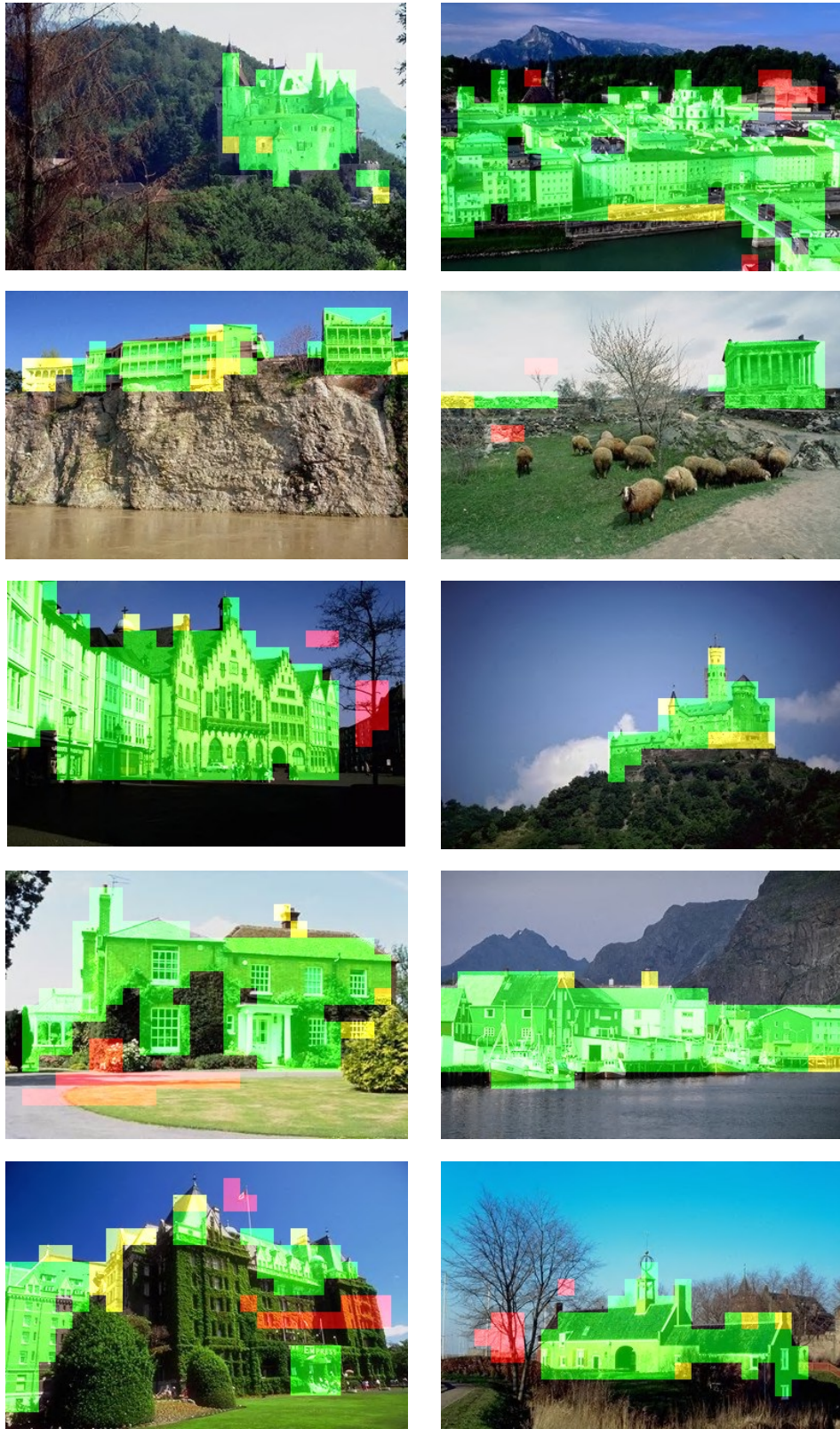


Fig. 4. Example results for structure detection task using a majority voting on a set of 15 CRF spanning trees with randomly generated structures. Spanning tree structures used during learning are not memorized; the same optimized set of parameters is used for all randomly generated spanning trees. True positives are highlighted in green, false positives are in red, false negatives are in yellow.

approximations based on spanning trees have shown better detection rate and precision than the ones produced by the original model with LBP inference.

The performance of the investigated approximations that do not memorize tree structure is slightly better than that of the model with memorized structure of the spanning trees. We hypothesize that the robustness towards edge removal in the original loopy model in the process of tree construction and towards the possibility of presence of the same edge in multiple trees is the result of combination of two factors. The first is that different spanning trees in ensemble provide alternative pathways between node clusters of the original graph model. This compensates for removed edges in the original model. The second is the use of the same set of tree parameters,  $\theta^*$ , that is optimized for a variety of randomly generated spanning trees. This leads to a parameter template that is more robust towards missing or duplicated edges.

#### IV. DISCUSSION

We have investigated an approximation approach to learning and inference in CRF using a decomposition of the original grid-structured graph based on a set of randomly sampled spanning trees. The results on two synthetic and real-world imagery datasets have demonstrated better performance of the investigated approximation approach than the performance of the original grid-structured model with LBP probabilistic inference. The interesting finding is that the approach that does not memorize the structure of spanning trees has shown slightly better performance than the model with the memorized tree structures. This is similar to the finding in [25] that investigated an approximation based on a cascade of spanning trees. This finding will be the subject of further investigation as well as approximations based on the trees with thicker treewidth (e.g., [5, 6, 7, 8]), which are still computationally tractable.

#### ACKNOWLEDGMENT

This work was supported by the U.S. Department of Energy through the LANL LDRD Program.

#### REFERENCES

- [1] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: probabilistic models for segmenting and labeling sequence data," In: Proceedings of the 18th International Conference on Machine Learning (ICML), 2001.
- [2] J.M. Hammersley and P. Clifford, "Markov fields on finite graphs," unpublished, 1971.
- [3] J. Besag, "Spatial interactions and the statistical analysis of lattice systems," *Journal of the Royal Statistical Society, Series B*, 36(2):192-236, 1974.
- [4] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1998.
- [5] F.R. Bach and M.I. Jordan, "Thin junction trees," in *Advances in Neural Information Processing Systems (NIPS)*, 2001.
- [6] A. Globerson and T. Jaakkola, "Approximate inference using planar graph decomposition," in *Advances in Neural Information Processing Systems (NIPS)*, 2006.
- [7] N.N. Schraudolph and D. Kamenetsky, "Efficient exact inference in planar Ising models," in *Advances in Neural Information Processing Systems (NIPS)*, 2008.
- [8] D. Batra, A.C. Gallagher, D. Rarikh, and T. Chen, "Beyond trees: MRF inference via outer-panar decomposition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [9] V. Kolmogorov and R. Zabih, "What energy functions can be optimized via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147-159, 2004.
- [10] P. Dagum and M. Luby, "Approximating probabilistic inference in Bayesian belief networks is NP-hard," *Artificial Intelligence*, 60:141-153, 1993.
- [11] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222-1239, 2001.
- [12] S.E. Shimony, "Finding MAPs for belief networks is NP-hard," *Artificial Intelligence*, 68: 399-410, 1994.
- [13] J. Besag, "Statistical analysis of non-lattice data," *The Statistician*, 24:179-195, 1975.
- [14] J. Besag, "Efficiency of pseudo-likelihood estimation for simple gaussian fields," *Biometrika*, 64(3):616-618, 1977.
- [15] C. Sutton and A. McCallum, "Piecewise pseudolikelihood for efficient CRF training," in *Proceedings of the 24th International Conference on Machine Learning (ICML)*, 2007.
- [16] C. Sutton and A. McCallum, "Piecewise training for structured prediction," *Machine Learning*, 77(2-3):165-194, 2009.
- [17] A.U. Asuncion, Q. Liu, A.T. Ihler, and P. Smyth, "Learning with blocks: composite likelihood and contrastive divergence," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.
- [18] M. Wainwright, T. Jaakkola, and A. Willsky, "MAP estimation via agreement on (hyper)trees: message-passing and linear programming approaches," *IEEE Transactions on Information Theory*, 51(11):3697-3717, 2005.
- [19] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568-1583, 2006.
- [20] P. Pletscher, C.S. Ong, and J.M. Buhmann, "Spanning tree approximations for conditional random fields," in *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.
- [21] D.B. Wilson, "Generating random spanning trees more quickly than the cover time," in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, 1996.
- [22] D.C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization methods," *Mathematical Programming*, 45:503-528, 1989.
- [23] S. Kumar and M. Hebert, "Discriminative fields for modeling spatial dependencies in natural images," in *Advances in Neural Information Processing Systems (NIPS)*, 2003.
- [24] S. Kumar and M. Hebert, "Discriminative random fields," *International Journal of Computer Vision*, 68(2):179-201, 2006.
- [25] A.N. Skurikhin, "Hierarchical spanning tree-structured approximation for conditional random fields: an empirical study," *Lecture Notes in Computer Science (LNCS)*, 8888:85-94, 2014.